Computing Systems

Introduction to Operating Systems

Contents

- The History of Operating Systems
- Operating System Architecture
- Coordinating the Machine’s Activities
- Handling Competition Among Processes
- Security

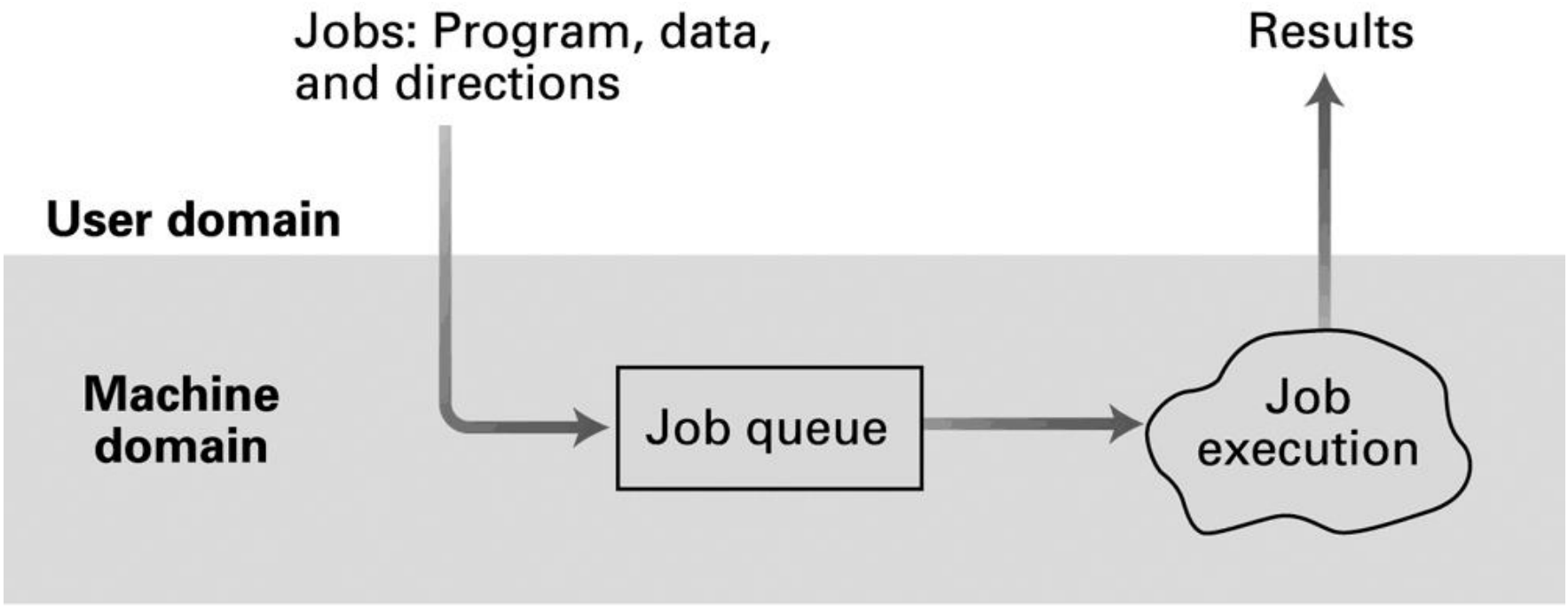
Functions of Operating Systems

- Oversee operation of computer
- Store and retrieve files
- Schedule programs for execution
- Coordinate the execution of programs

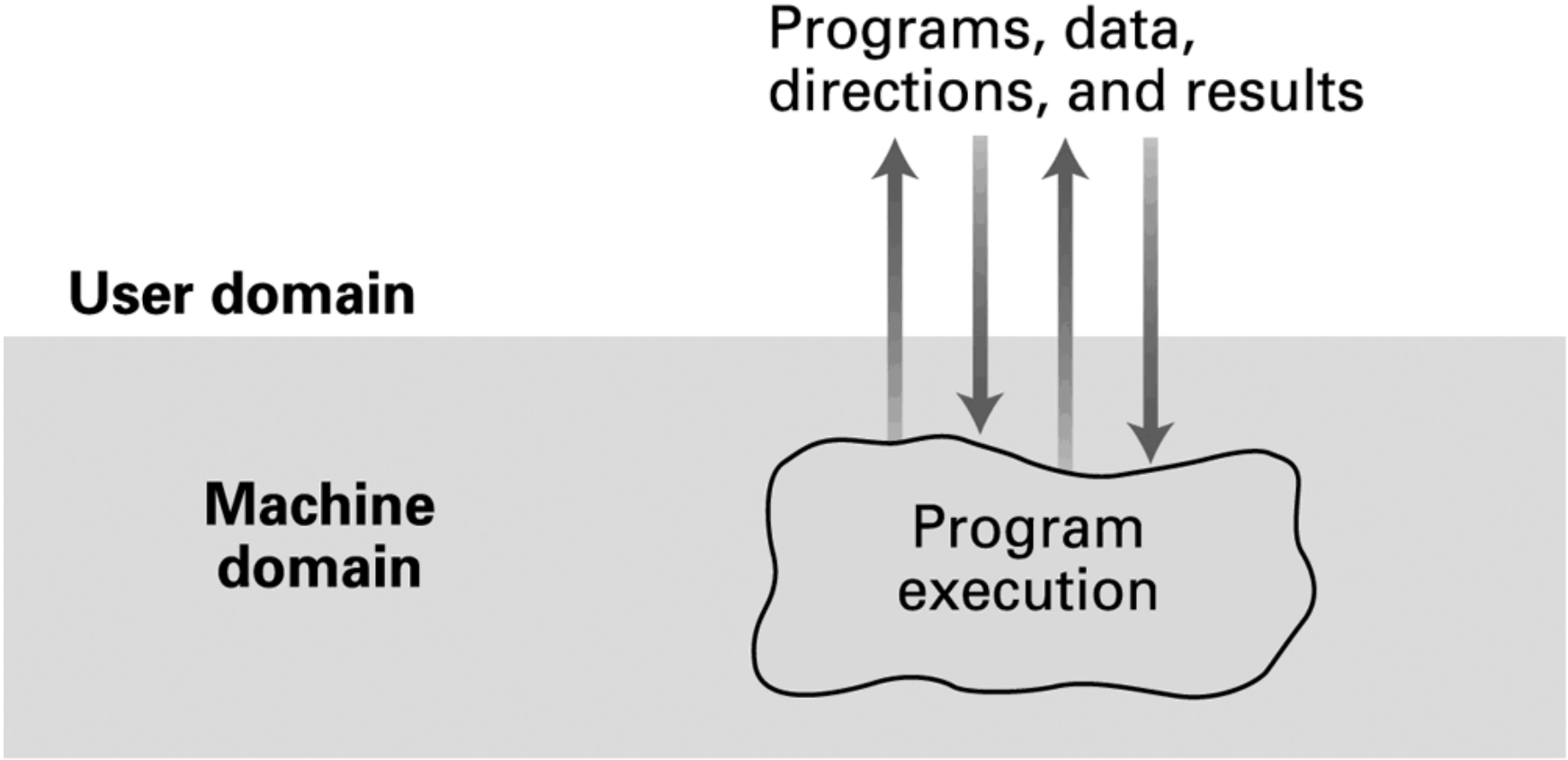
Evolution of Shared Computing

- Batch processing
- Interactive processing
  - Requires real-time processing
- Time-sharing/Multitasking
  - Implemented by Multiprogramming
- Multiprocessor machines

Batch processing



Interactive processing



Time Sharing / Multitasking

- Users seeking services from same machine at the same time – time sharing
  - Implemented using a technique called multiprogramming (time is divided into multiple intervals, execution of one job is limited to a single time interval)
- Multiple terminals connected to same machine
  - Driven by the fact that in the past computers were very expensive
- When multiprogramming is applied to single-user environments is usually called multitasking

Multiprocessor Operating Systems

- Provide time sharing/multi-tasking capabilities by assigning different tasks to different processors as well as sharing the time of one single processor
- Problems to solve:
  - Load balancing – dynamically allocating tasks to the various processor so that all of them are used efficiently
  - Scaling – breaking tasks into sub-tasks compatible with the number of processors available
- Trend to develop a network wide operating system rather than networks of individual operating systems

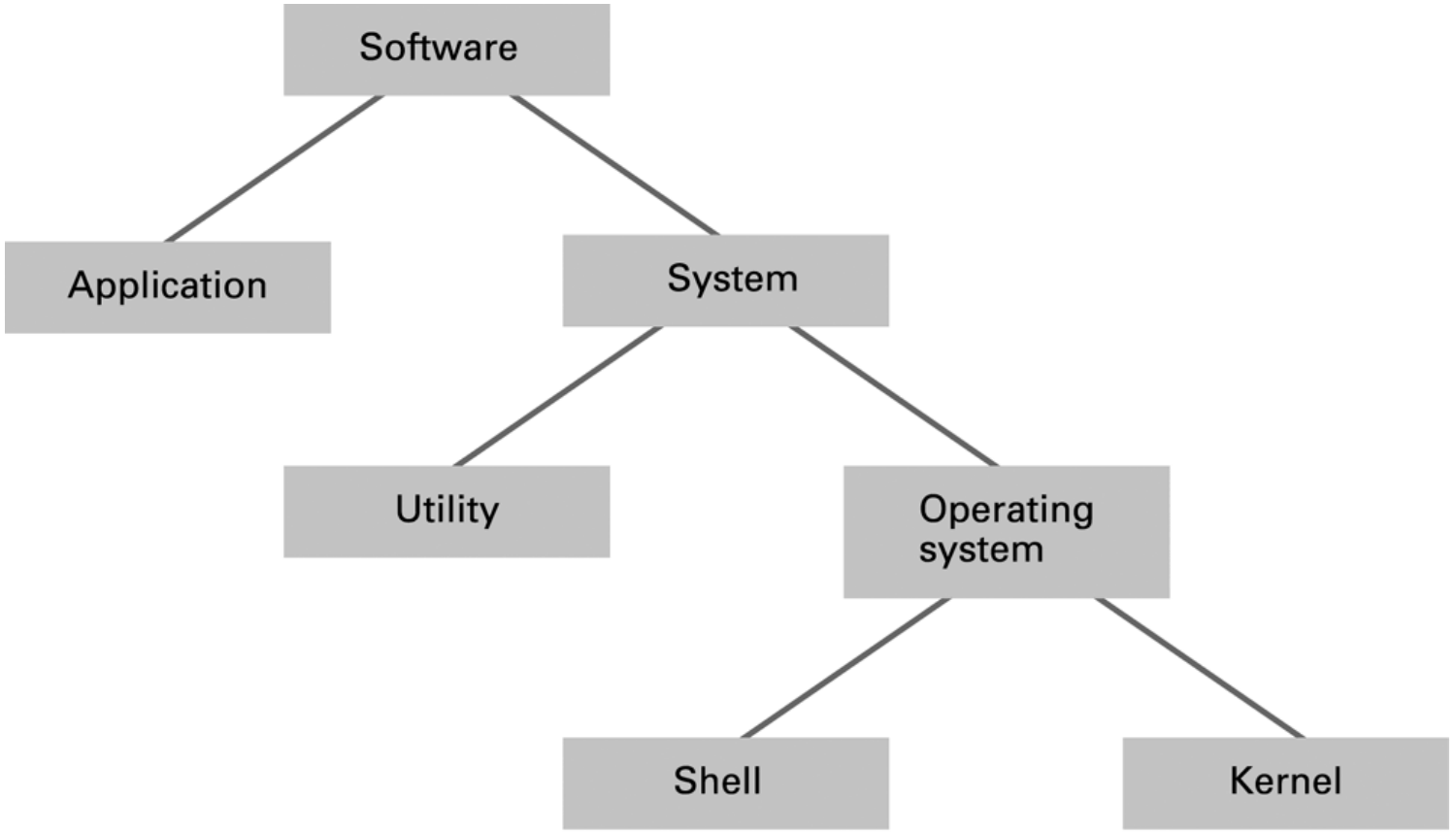
Embedded Operating Systems

- Used in hand held devices, mobile phones, cars, etc…
- Limited data storage and power conservation are the big challenges
- Examples: VxWorks, Windows CE (Pocket PC), Palm OS, Symbinan, ThredX, RomDOS, etc…

Types of Software

- Application software
  - Performs specific tasks for users: spreadsheets, database systems, desktop publishing, program development, games, etc…
- System software
  - Provides infrastructure for application software
  - Consists of **operating system** and **utility software**

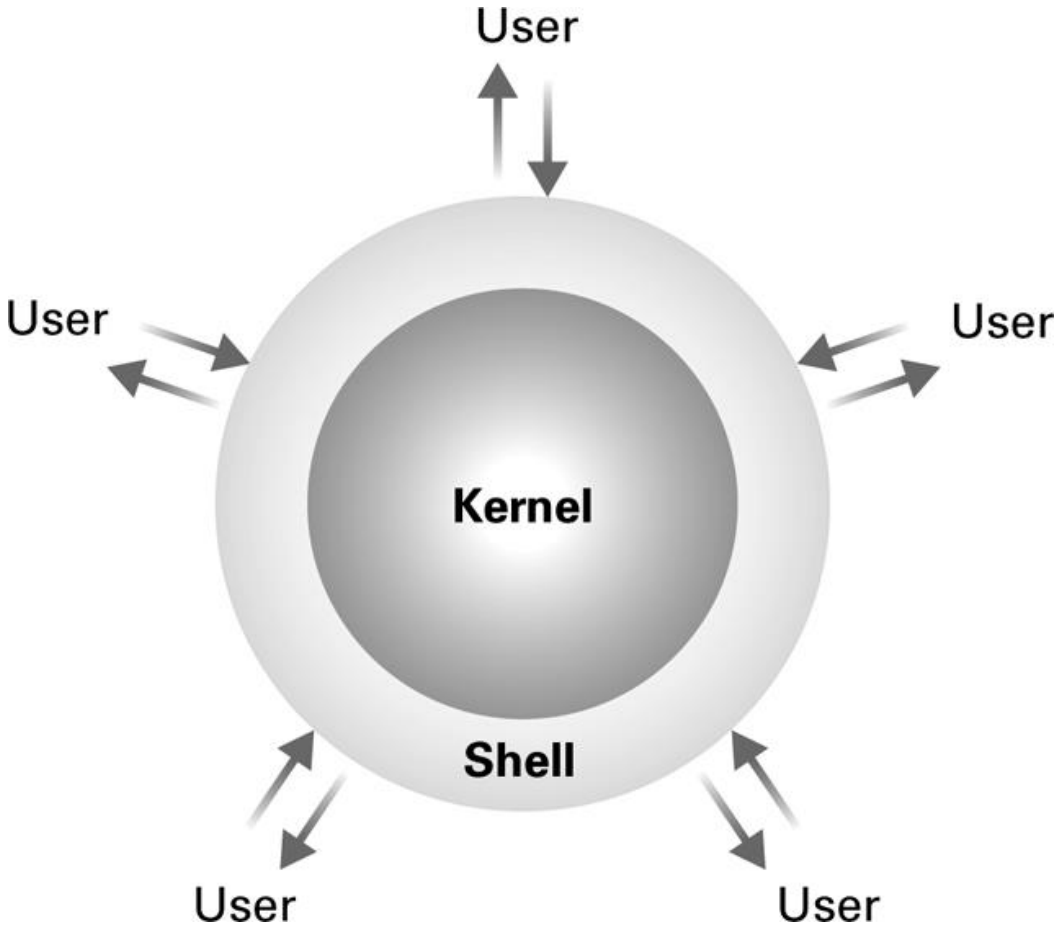
Software classification



Operating System Components

- **Shell:** Communicates with users
  - Text based
  - Graphical user interface (GUI)
- **Kernel:** Performs basic functions
  - File manager
  - Device drivers
  - Memory manager
  - Process manager (Scheduler, dispatcher, etc..)

The shell as an interface between users and the operating system



File Manager

- Role – coordinate the use of machine's mass storage facilities
- Hierarchical organization
  - **Directory** (or **Folder**): A user-created bundle of files and other directories (subdirectories)
  - **Directory Path:** A sequence of directories within directories
- Access/operations to files is provided by file manager via a **file descriptor**

Device Manager

- Part of OS presented as a collection of device drivers – specialized software that communicate with the controllers to carry out operations on peripheral devices connected to the computer
- Each driver is specifically designed for its type of device (e.g. printer, monitor, etc..) and translates generic requests into device specific sequence of operations

Memory Manager

- Has the task of coordinating the use of main memory – allocates/deallocates space in main memory
- When the total required memory space exceeds the physical available space.
  - May create the illusion that the machine has more memory than it actually does (**virtual memory**) by playing a “shell game” in which blocks of data (**pages**) are shifted back and forth between main memory and mass storage

Processes

- **Process:** The activity of executing a program (NOT THE SAME THING AS A PROGRAM!!!)
  - Program – static set of directions (instructions)
  - Process – dynamic entity whose properties change as time progresses. It is an instance in execution of a program.
- **Process State:** Current status of the activity
  - Program counter
  - General purpose registers
  - Related portion of main memory

Process Manager

- **Scheduler** – the part of kernel in charge with the strategy for allocation/de-allocation of the CPU to each competing process
  - Maintains a record of all processes in the OS (via a **process table**), introduces new processes to this pool and removes the ones that completed
- **Dispatcher** is the component of the kernel that overseas the execution of the scheduled processes
  - Achieved by multiprogramming

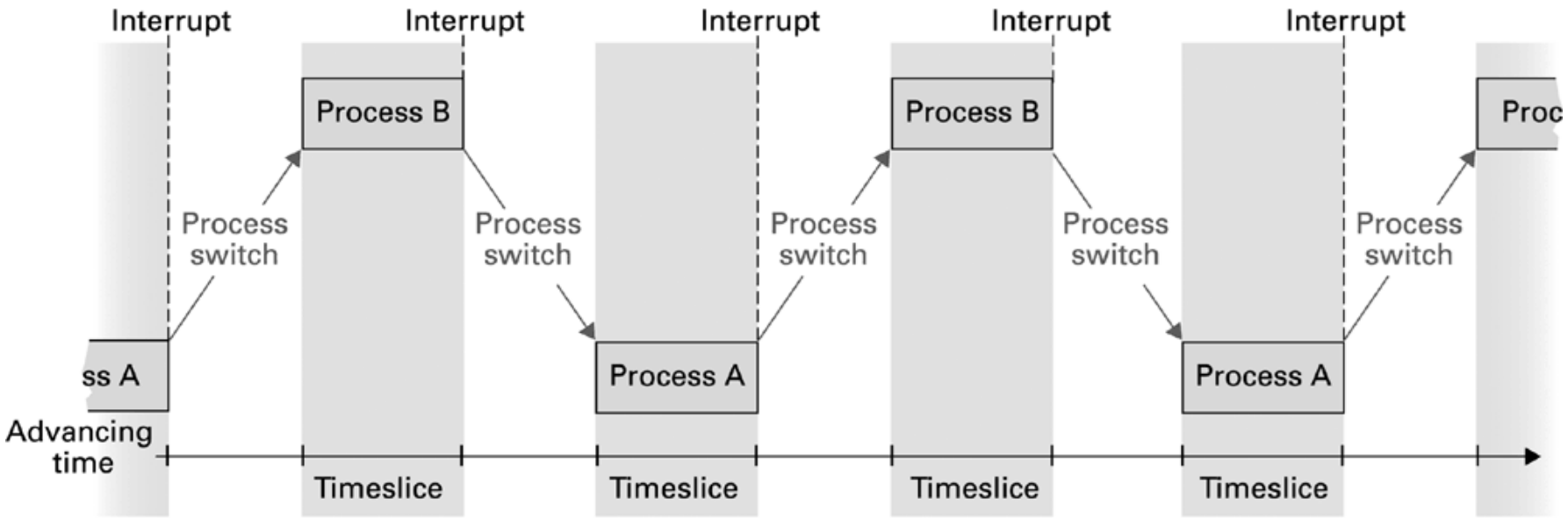
Scheduler

- **Scheduler:** Adds new processes to the process table and removes completed processes from the process table
- Process table contains
  - Memory area assigned to the process
  - Priority of the process
  - State of the process (ready or waiting)

Dispatcher

- **Dispatcher:** Controls the allocation of CPU (of time slices) to the processes in the process table
  - The end of a time slice is signaled by an interrupt.
  - Each process is allowed to execute for one time slice
- It performs "**process switch**" – procedure to change from one process to another
  - ProcessA→ Dispatcher→ ProcessB

Time-sharing between process A and process B



Handling Competition for Resources

Important task of OS is to allocate *resources* to the processes

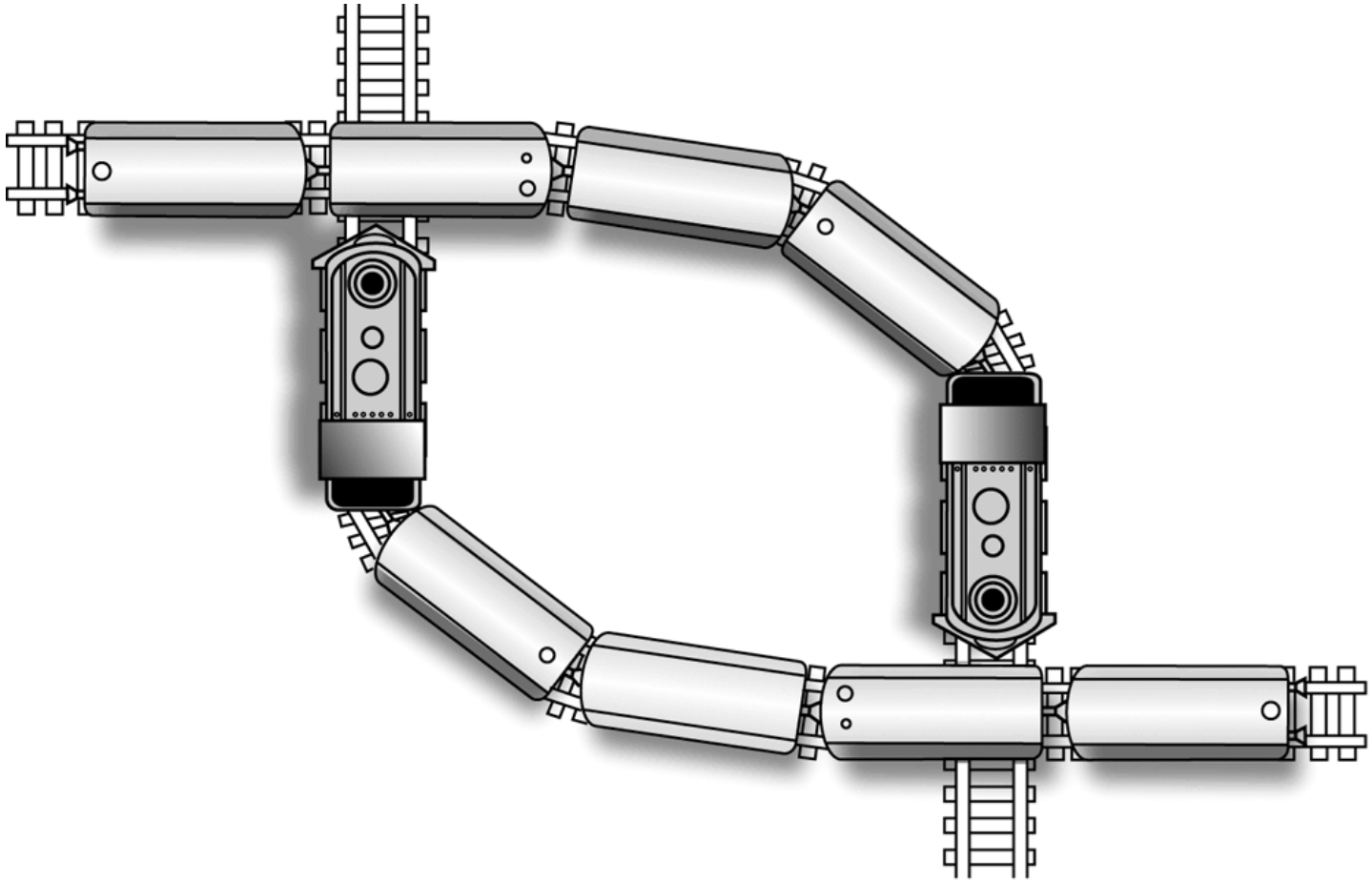
- **Semaphore:** A “control flag”
- **Mutual exclusion:** Requirement for proper implementation of a critical region so that only one process at a time will execute the sequence of instructions part of a critical region

Deadlock

Another problem of resource allocation :-

- Processes block each other from continuing
- Conditions required for deadlock
  1. Competition for non-sharable resources
  2. Resources requested on a partial basis
  3. An allocated resource can not be forcibly retrieved

A deadlock resulting from competition for nonshareable railroad intersections



Security

- One of the role of OS is to provide security
- Attacks from outside
  - Insecure passwords
  - Counter measures
    - Auditing software

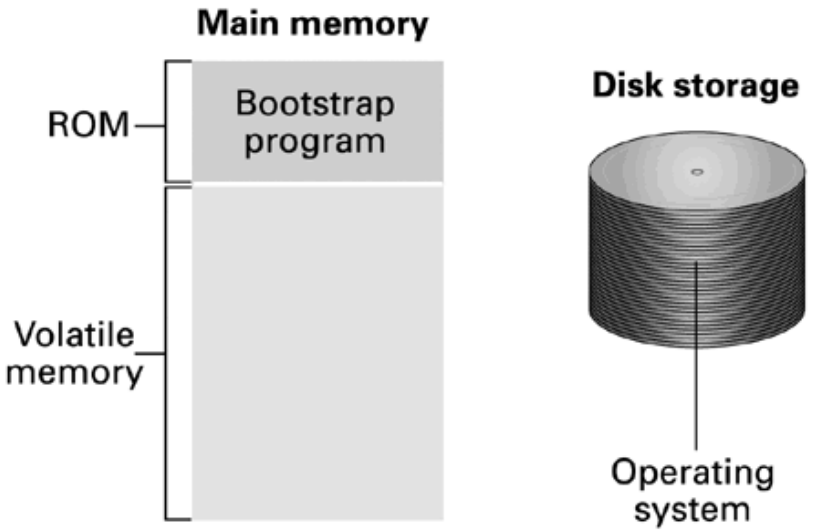
Security (continued)

- Attacks from within
  - Counter measures: Control process activities via privileged modes and privileged instructions
  - Examples on attacker SW:
    - Alters the timer of OS – extend its own time slice and dominate the machine
    - Access to peripheral devices directly – access to files that access would have been denied
    - Access memory cells outside its allowed area, it can read and alter data from other processes

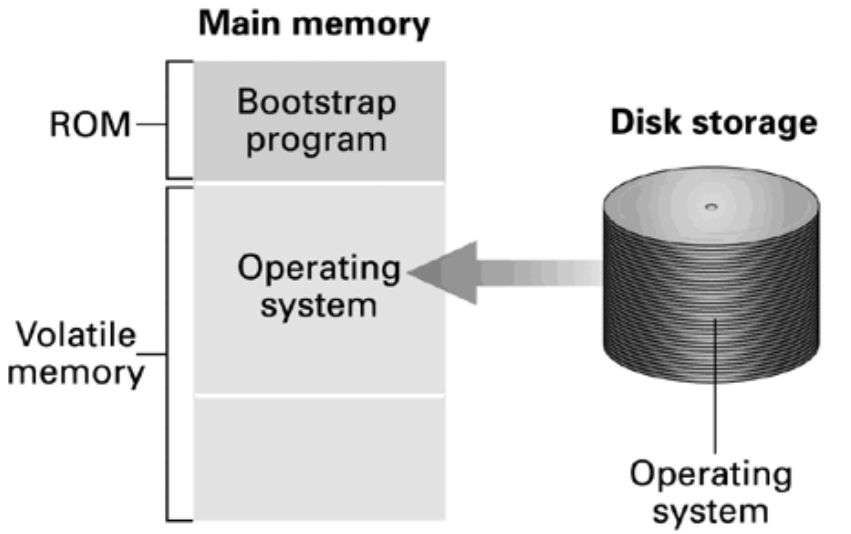
Getting OS Started (Bootstrapping)

- **Booting:** Procedure that transfers the OS from mass storage (permanent) into the main memory (volatile-thus empty when machine is turned on)
- **Bootstrap:** Program in ROM (example of firmware)
  - Run by the CPU when power is turned on (PC starts at pre-defined address when power is applied)
  - Transfers operating system from mass storage to main memory
  - Executes jump to operating system

The booting process



Step 1: Machine starts by executing the bootstrap program already in memory. Operating system is stored in mass storage.



Step 2: Bootstrap program directs the transfer of the operating system into main memory and then transfers control to it.